P RTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: [✓ The ACM Digital Library] The Guide

"dynamic scoping" and tcl SEARCH

THE ACM DIGITAL LIBRARY

Terms used **dynamic scoping** and **tcl**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Results 1 - 20 of 200

Best 200 shown

Result page: **1** 2 3 4 5 6 7 8 9 1

1 Dynamic variables
David R. Hanson, Todd A. Proebsting
May 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2001 confer**
Issue 5
Publisher: ACM Press
Full text available: pdf(943.02 KB) Additional Information: full citation, abst

Most programming languages use static scope rules for associating uses of identifiers with their
efficiently. Some popular languages—Perl, Tel, TeX, and Postscript—offer dynamic scope, becau
example. Programmers must simulate dynamic scope to implement this kind of usage in statical

2 Using aspectC to improve the modularity of path-specific customization in operating systen
Yvonne Coady, Gregor Kiczales, Mike Feeley, Greg Smolyn
September 2001 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 8th Eu**
international symposium on Foundations of software engineering ESE
Publisher: ACM Press
Full text available: pdf(109.16 KB) Additional Information: full citation, abst

Layered architecture in operating system code is often compromised by execution path-specific
specific customizations are difficult to modularize in a layered architecture because they involve
slices through the layers. An initial experiment using an aspect-oriented programming language

Keywords: aspect-oriented programming, operating system design, software modularity

3 Implicit context: easing software evolution and reuse
Robert J. Walker, Gail C. Murphy
November 2000 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 8th AC**
twenty-first century applications SIGSOFT '00/FSE-8, Volume 25 Issue 6
Publisher: ACM Press
Full text available: pdf(1.24 MB) Additional Information: full citation, abst

Software systems should consist of simple, conceptually clean software components interacting
end up interacting for reasons unrelated to the functionality they provide. We refer to knowledg
component as extraneous embedded knowledge (EEK). EEK creeps into a system in many forms

Keywords: EEK, call history, contextual dispatch, extraneous embedded knowledge, flexibility,

4 Parsing and evaluation of APL with operators